

### AMENDMENTS TO THE CLAIMS

Please amend the claims as follows.

1. (Currently Amended) ~~A Method~~ method for protecting a portable object against denial of service type attacks, said portable object comprising a memory where at least one credential is stored, each credential being associated with a service, the method comprising ~~comprises~~ ~~the steps of:~~
  - verifying that an ~~[[the]]~~ entity requesting ~~[[a]]~~ the service is an authorized entity for accessing the service, based on an algorithm involving the at least one credential,
  - delivering the requested service only when the verification step succeeds~~[[ed]]~~,
  - blocking the at least one credential associated with the service after a certain number of verification steps fail~~[[ed]]~~, and
  - ~~wherein, when if~~ the verification step fails~~[[ed]]~~, ~~the method further comprises the steps of:~~
    - waiting ~~during~~ for a waiting duration before allowing a new verification step.
2. (Currently Amended) ~~The Method~~ method ~~for protecting a portable object against denial of service type attacks~~ as recited in claim 1, wherein the waiting duration is constant for each failed verification step.
3. (Currently Amended) ~~The Method~~ method ~~for protecting a portable object against denial of service type attacks~~ as recited in claim 1, wherein the waiting duration is variable for each failed verification step.
4. (Currently Amended) ~~The Method~~ method ~~for protecting a portable object against denial of service type attacks~~ as recited in claim 1 ~~one of the previous claims~~, wherein the waiting duration is equal to zero for a first predetermined number of failed verification steps, and the waiting duration is greater than zero for a second predetermined number of failed verification steps.

5. (Currently Amended) ~~The Method~~ method ~~for protecting a portable object against denial of service type attacks~~ as recited in claim 1 ~~one of the previous claims~~, wherein the method ~~step of waiting during a duration~~ is resumed prior to the waiting duration elapsing ~~if in case said step is interrupted before the duration has elapsed~~.
6. (Currently Amended) ~~The Method~~ method ~~for protecting a portable object against denial of service type attacks~~ as recited in claim 1, wherein ~~said~~ the ~~at least one~~ credential is one selected from the group consisting of a personal identification number, [[or]] a key, and [[or]] a code.
7. (Currently Amended) ~~The Method~~ method ~~for protecting a portable object against denial of service type attacks~~ as recited in claim 1, wherein ~~said~~ the ~~entity is~~ at least one selected from the group consisting of a user, [[or]] a terminal, [[or]] a server, and [[or]] an application.
8. (Currently Amended) ~~The Method~~ method ~~Method for protecting a portable object against denial of service type attacks~~ as recited in claim 1 ~~further comprising one of the previous claims, wherein the method further comprises the steps of:~~
- decrementing a counter associated with the at least one credential each time [[a]] the verification step is performed, said counter having values ranging between an initial value and a credential blocking value,
- resetting the counter~~[[s]]~~ to the initial value when the verification step succeeds, and,
- if when the counter has reaches~~[[d]]~~ an intermediate value, ~~the method further comprises the steps of:~~
- waiting for ~~during a~~ the waiting duration when verification step fails~~[[ed]]~~,
- blocking the at least one credential when the counter reaches the credential blocking value,
- wherein the intermediate value is between the initial value and the credential blocking value.

9. (Currently Amended) ~~The Method method for protecting a portable object against denial of service type attacks as recited in claim 1 one of the claims 1 to 7, wherein the method further comprises the steps of~~ further comprising:
- decrementing a first counter associated with the at least one credential each time ~~[[a]]~~ the verification step is performed, ~~said the~~ the first counter having values ranging between a first initial value and an intermediate value,
  - and, when the first counter ~~has reaches~~ reaches ~~[[d]]~~ the intermediate value, ~~the method further comprises the steps of:~~
  - decrementing a second counter associated with the first counter, ~~said the~~ the second counter having values ranging between a second initial value and a credential blocking value,
  - resetting the first counter to the first initial value and the second counter to the second initial value ~~if when~~ when verification step succeeds,
  - waiting during a duration ~~if when~~ when verification step fails ~~[[ed]]~~, and
  - blocking the credential ~~if when~~ when the second counter reaches the credential blocking value.
10. (Currently Amended) ~~The Method method for protecting a portable object against denial of service type attacks as recited in one of the previous claims as recited in claim 1, wherein, when verification step failed, waiting during the duration the step of waiting during a determined duration is implemented by a~~ comprises using a waiting loop mechanism.
11. (Currently Amended) ~~The Method method for protecting a portable object against denial of service type attacks as recited in claim 8, wherein the decrementing step of the counter or the first counter or the second counter are~~ is performed before the verification step.
12. (Currently Amended) ~~The Method method for protecting a portable object against denial of service type attacks as recited in claim 8, wherein the decrementing step of the counter or the first counter or the second counter are~~ is performed after the verification step.

13. (Currently Amended) ~~A Portable~~ portable object comprising, in particular a smart card,  
comprises:

a memory wherein at least one credential is stored, each wherein the at least one  
credential being is associated with a service, the at least one said credential being  
is used to verify that the an entity requesting [[a]] the service is an authorized  
entity for accessing the service,

a counter associated with the at least one credential which is decremented each time a  
verification that the entity requesting [[a]] the service is [[an]] the authorized  
entity for accessing the service fails[[ed]], ~~said the~~ counter having values ranging  
between an initial value and a credential blocking value, ~~said the~~ counter being  
reset to the initial value when verification succeeds, and

wherein ~~said the~~ portable object further comprises:

a waiting loop mechanism which is activated when the counter has reached a  
intermediate value and each time [[a]] the verification fails[[ed]].

14. (Currently Amended) ~~The Portable~~ portable object, as recited in claim 13 the previous claim,  
wherein the counter comprises a first counter and a second counter, the first counter  
associated with the at least one credential being decremented each time the [[a]] verification  
that the entity requesting the [[a]] service is the [[an]] authorized entity for accessing the  
service fails[[ed]], ~~said the~~ first counter having values ranging between a first initial value  
and [[an]] the intermediate value, the second counter being decremented when the first  
counter has reached the intermediate value and each time the [[a]] verification that the entity  
requesting the [[a]] service is the [[an]] authorized entity for accessing the service fails[[ed]],  
the said second counter having values ranging between a second initial value and a credential  
blocking value.

15. (Currently Amended) ~~The Portable~~ portable object, as recited in claim 13 one of the claims  
13 to 14, wherein the waiting loop mechanism comprises a loop flag used to resume the step  
of waiting during [[a]] the duration if the portable object is performed by the waiting loop  
mechanism in case said step is interrupted before the duration has elapsed.

16. (Currently Amended) A computer program product comprising a computer readable medium, having thereon computer program code means, when said program is loaded into the memory of the portable object, to make the portable object execute the method for protecting said portable object against denial of service type attacks as recited in claim 1 ~~any of the claims 1 to 12.~~